Using A Neural Network To Identify Banknote Forgeries

Banknote forgeries are harmful to banks given that:

1. Monetary losses for banks
2. The real value of currency at large is lowered
3. They can be passed around unknowingly, potentially implicating innocent people

Therefore, it is useful for banks to identify forgeries before they can become circulated in order to avoid the three mentioned pitfalls and potentially trace back the culprit responsible.

Using the source below, the dataset can be downloaded:

Source: https://archive.ics.uci.edu/dataset/267/banknote+authentication

Citation: Lohweg,Volker. (2013). banknote authentication. UCI Machine Learning Repository. https://doi.org/10.24432/C55P57.

The dataset contains information on 1372 grayscale pictures of banknotes. Each image is 400 x 400 pixels large.

The target variable is labelled "Class" where a value of 1 indicates the note is a forgery and a value of 0 indicates the note is legitimate.

The explanatory variables are the variance, skewness, kurtosis, and entropy of the pixel values of the grayscale images. A pixel in grayscale format is given a value ranging from 0 to 255 where 0 is all black and 255 is all white. By studying the pixel values using the grayscale format range, we can more easily detect forgeries given that legitimate banknotes may have very slight variations in designs, but forgeries will tend to have greater variations in design.

Here are the definitions on the four statistical terms for the explanatory variables:

1. Variance refers to the spread of the pixel values. High variance would see a greater number of pixel values being present in the dataset while lower variance would see a lower number of pixels in the dataset.
2. Skewness refers to the symmetricity of the pixel values where the data can be skewed to the right or left to some degree. A positive skewness value indicates skewness to the right, a negative skewness value indicates skewness to the left, and zero would be perfectly symmetrical.
3. Kurtosis refers to the sharpness of the peak of the distribution. Negative kurtosis would mean more data is centered around the mean than the tails and positive kurtosis would mean more data is contained near the tails than the mean.
4. Entropy refers to the prescence of outliers in the data where positive entropy has more outliers and negative entropy has less outliers.

```r
# Loads the neuralnet library
library(neuralnet)

# Reads in the dataset and gives the columns names
data = read.delim("/Users/rolandgraham/Documents/data_banknote_authentication.txt", header = FALSE, sep
colnames(data)[1] = "variance"
colnames(data)[2] = "skewness"
colnames(data)[3] = "kurtosis"
colnames(data)[4] = "entropy"
colnames(data)[5] = "class"

# Standardizes the explanatory variables to assure they are measured on the
# same scale
```

```
data$variance = (data$variance - min(data$variance)) / (max(data$variance) - min(data$variance))
data$skewness = (data$skewness - min(data$skewness)) / (max(data$skewness) - min(data$skewness))
data$kurtosis = (data$kurtosis - min(data$kurtosis)) / (max(data$kurtosis) - min(data$kurtosis))
data$entropy = (data$entropy - min(data$entropy)) / (max(data$entropy) - min(data$entropy))
```

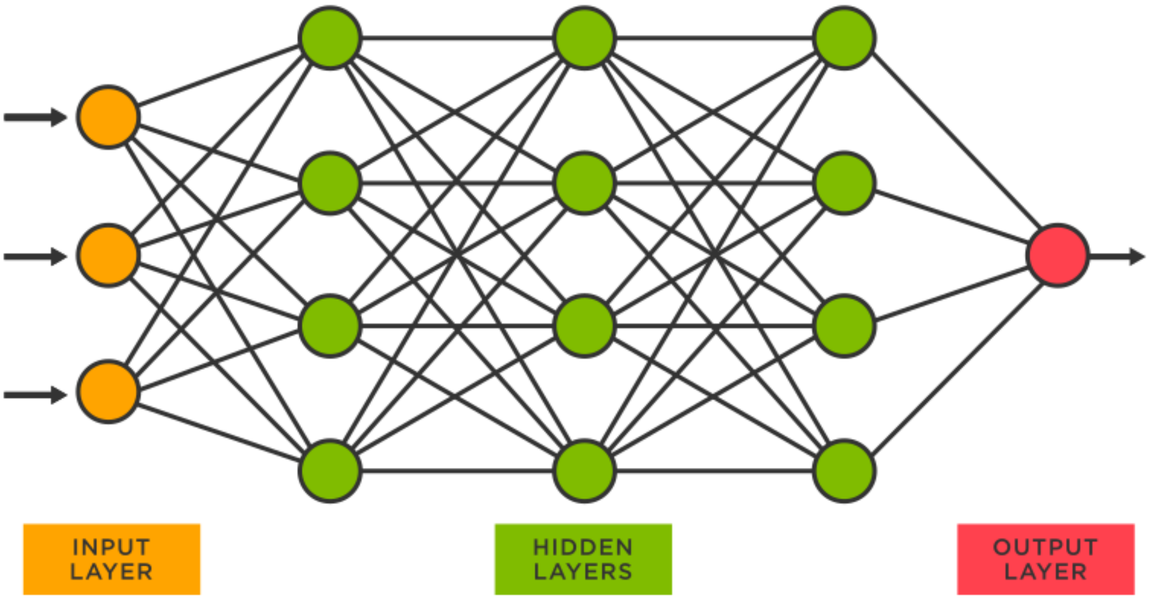
Now I will explain the purpose of neural networks and how we can use these four features to predict whether a banknote is a forgery or not.

Below is the general structure of a neural network:

```
library(cowplot)
library(magick)
```

```
## Linking to ImageMagick 6.9.12.3
## Enabled features: cairo, fontconfig, freetype, heic, lcms, pango, raw, rsvg, webp
## Disabled features: fftw, ghostscript, x11
```

```
fig_svg<-cowplot::ggdraw()+cowplot::draw_image("neural.svg")
plot(fig_svg)
```



The goal of neural networks is to classify observations into groups. In this case, is the banknote a forgery or legitimate? This will be accomplished through the four features discussed earlier.

You can see that there are three main parts to a neural network: the input layer, the hidden layer, and the output layer. Each circle represents a node and the number of nodes in each layer will be important in designing the neural network. The input layer will be represented by four features we will use to predict the forgeries, so in our neural network there will be four nodes in the input layer. The output layer will be represented by the variable we are predicting, in this case, the class variable. So in our neural network, there will be one node in the output layer.

The aspect of the neural network we will have to design ourselves, however, is how many levels will the hidden layer have and how many nodes will be on each level. In the example shown above, the hidden layer has three levels with four nodes each, however we can adjust those numbers for our neural network.

Ideally, the number of levels should be either one or two for data with a low amount of explanatory variables and between three to five for data with a high amount of explanatory variables. Given that we are working with only four variables, this gives us either one or two levels to work with.

And generally speaking, the number of nodes on the hidden layers should be a value between the number of nodes on the output layer and that of the input layer. This means each level should have between one to four nodes.

These two rules makes sure the neural network does not become too complex, which will lead to the neural network to being overtrained. Overtraining means that the neural network will be too fitted to the specific patterns of the training dataset and will be inapplicable identifying forgeries in other datasets.

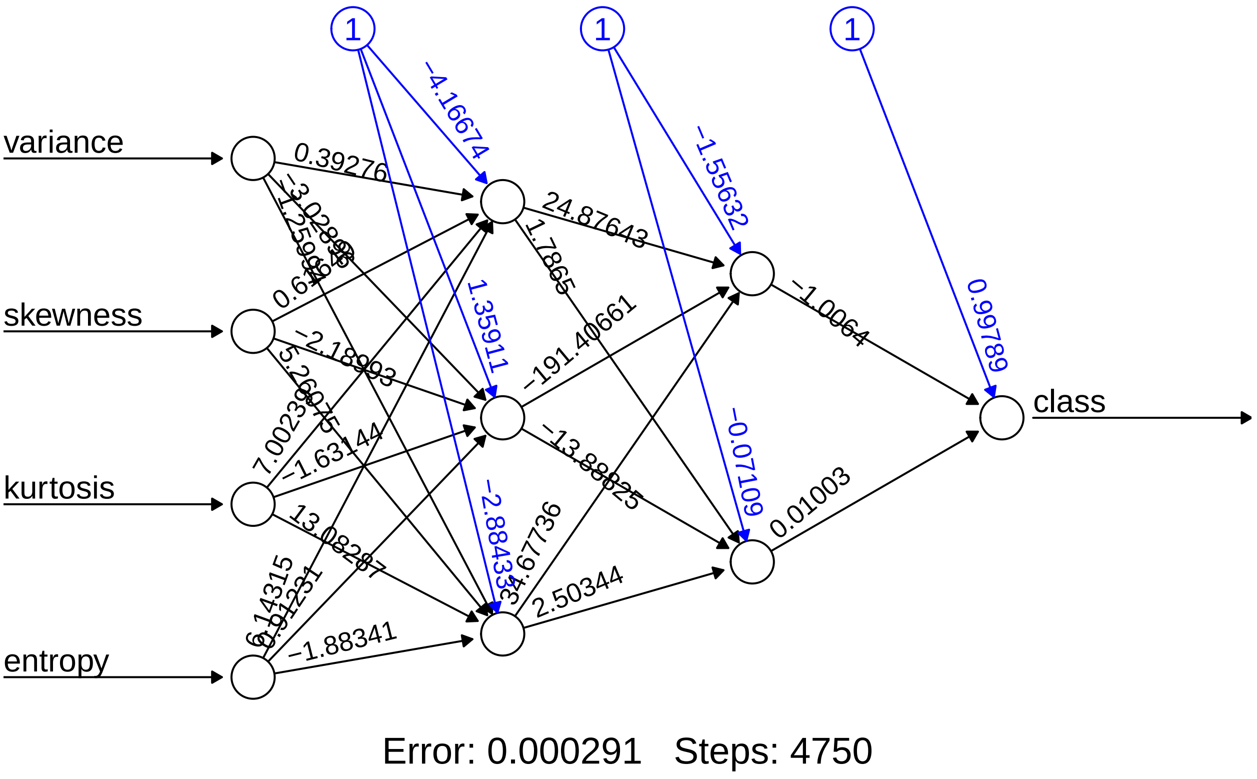
Given that we only have sixteen combinations of hidden layers (1-2) and hidden nodes (1-4) for our neural network, it is alright to test out each combination and select the one that results in the lowest error.

I reran the below code several times with the different combinations. The combination that resulted in the lowest error was two hidden layer levels with three nodes on the first level and two nodes on the second. Below is the resulting plot.

```
# Shuffles the dataset so the training and testing segments get a mix of
# positive and negative responses for forgeries
set.seed(35)
shuffled.data = data[sample(1:nrow(data)), ]

# Divides the dataset into training and testing segments
set.seed(35)
sample <- sample(c(TRUE, FALSE), nrow(shuffled.data), replace=TRUE, prob=c(0.7,0.3))
train  <- shuffled.data[sample, ]
test   <- shuffled.data[!sample, ]

# Creates and plots the neural network with two hidden layers and three
# hidden nodes
set.seed(35)
net = neuralnet(class ~ ., train, hidden = c(3,2))
plot(net, rep="best")
```



The black numbers represent the weights attached to each node pathway and the blue numbers represent the bias attached to each node. These numbers will not be of particular use to us in understanding the neural network given how complex the calculations are. One con with neural networks is that they are essentially black boxes where we understand what goes in and out of the system, but not how the hidden layers transform the input layers to get to the output layer. The way to measure how accurate our neural network is is to create a confusion matrix of the testing dataset.

```
# Cross-validates the neural net on the testing data and creates a confusion
# matrix
pred = compute(net, test)
cross.val = data.frame(actual = test$class, prediction = pred$net.result)
cross.val.round = sapply(cross.val, round, digits = 1)
cross.val.round.df = data.frame(cross.val.round)
attach(cross.val.round.df)

# Prints the confusion matrix
table(actual, prediction)
```

```
##       prediction
## actual   0   1
##      0 231   0
##      1   0 202
```

The class values for the testing dataset were rounded to the nearest tenth of a decimal where a value of 0.9 or higher was treated as a forgery and a value of 0.1 or lower was treated as legitimate. Given that the neural network as able to accurately predict all observations demonstrates that it is very accurate.

This neural network could then potentially be run through many other sets of data in order to determine forgeries from legitimate banknotes given that the banknote images are similar to what was used in the training and testing datasets.